UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/028,899	12/20/2001	Robert Alan Reid	01 P 09444 US	2160

21171 7590 09/24/2007

STAAS & HALSEY LLP
SUITE 700
1201 NEW YORK AVENUE, N.W.
WASHINGTON, DC 20005

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2195	
MAIL DATE	DELIVERY MODE
09/24/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/028,899	REID, ROBERT ALAN	
	Examiner	Art Unit	
	Lewis A. Bullock, Jr.	2195	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *09 July 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-9 and 11-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-9 and 11-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2. Claims 1-8, 11-17 and 19 are rejected under 35 U.S.C. 102(b) as being anticipated by FLECK (U.S. Patent 6,128,641).

As to claim 1, FLECK teaches a method of chained switching execution of data processing tasks (routines / interrupt handlers) comprising: a first data processing task (routine / interrupt handler) executing a call (return) to a task switching function (context switch control unit / instruction control unit); the task switching function selecting a return address (context / pointer to CSA) corresponding to a second data processing task (routine / interrupt handler); and the task switching function executing a return operation (return) to the second data processing task; (via returning from the now current CSA to a previously executing context) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28). Fleck also teaches the current context resides in the general purpose and program state registers of the processor and to save the current context and create a new one, the following steps can be performed by the processor hardware: a context save area is taken from the free list, the current context is stored into the save area, the save area is added to the head of the previous context list wherein the steps are

performed in connection with a function call, or with taking a trap or interrupt thereby allowing the called function or interrupt or trap handler free to modify the general registers and other processor state without destroying the context of the calling function or interrupt task (col. 1, line 60 – col. 2, line 6). To switch back from a called function or trap or interrupt handler and switch back to the previous context, the following steps are performed by the processor handler: the save area at the head of the previous context list is removed from that list, the current context is loaded from the save area just removed from the previous context list, and the save area is added to the free context save area list. These steps are performed as part of the function return instruction, or the instruction to return from an interrupt or trap handler. FLECK also teaches on figure 4 that the PCX has three contexts which would be associated with three previously executing tasks / interrupt handlers. Therefore, the teachings of FLECK teach an interrupt handler/task executing such that when it finishes it invokes a return operation and the context switch control unit switches back to the head of the PCX pointer, thereby switching context / loading the previously executing second task / interrupt handler, such that when it finishes it invokes a return operation and the context switch control unit switches back to the new head of the PCX pointer, thereby switching context / loading the now previously executing third task / interrupt handler wherein all of the tasks are different from one another (via one is a caller, one is the called, one is an interrupt handler and some of the tasks execute in different regions).

As to claim 11, FLECK teaches an apparatus for chained switching execution of a tasks (routines / interrupt handlers) on a data processor (data processing unit) comprising: a memory (memory) having a first storage location for storing a return address corresponding to a second task (via storing CSA on previous context save area list); an input (call) for receiving information indicative of instructions of a task switching function that has been called by the first task (via a return from an executing routine / interrupt handler); and a memory management apparatus (via the context switch control unit) coupled to the input and the memory, and responsive to the instruction information indicating a return instruction for moving the return address corresponding to the task from the first storage location to a register of the data processor (via performing a context switch thereby returning to a previous CSA) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28). Fleck also teaches the current context resides in the general purpose and program state registers of the processor and to save the current context and create a new one, the following steps can be performed by the processor hardware: a context save area is taken from the free list, the current context is stored into the save area, the save area is added to the head of the previous context list wherein the steps are performed in connection with a function call, or with taking a trap or interrupt thereby allowing the called function or interrupt or trap handler free to modify the general registers and other processor state without destroying the context of the calling function or interrupt task (col. 1, line 60 – col. 2, line 6). To switch back from a called function or trap or interrupt handler and switch back to the previous context, the following steps are

performed by the processor handler: the save area at the head of the previous context list is removed from that list, the current context is loaded from the save area just removed from the previous context list, and the save area is added to the free context save area list. These steps are performed as part of the function return instruction, or the instruction to return from an interrupt or trap handler. FLECK also teaches on figure 4 that the PCX has three contexts which would be associated with three previously executing tasks / interrupt handlers. Therefore, the teachings of FLECK teach an interrupt handler/task executing such that when it finishes it invokes a return operation and the context switch control unit switches back to the head of the PCX pointer, thereby switching context / loading the previously executing second task / interrupt handler, such that when it finishes it invokes a return operation and the context switch control unit switches back to the new head of the PCX pointer, thereby switching context / loading the now previously executing third task / interrupt handler wherein all of the tasks are different from one another (via one is a caller, one is the called, one is an interrupt handler and some of the tasks execute in different regions).

As to claim 17, FLECK teaches a data processing apparatus, comprising: a data processing portion for executing tasks (data processing unit); a task switcher (context switch control unit) coupled to the data processing portion for switching from execution of a first task (routine / interrupt handler) to execution of a second task (routine / interrupt handler) (via a return operation), the task switcher including a memory (memory) having a storage location for storing a return address corresponding to the

second task (via storing CSA on previous context save area list), and an input for receiving information indicative of instructions of a task switching function that has been called by the first task (via a return operation to return execution to the previously executing context of a task or interrupt handler); a register coupled to the task switcher (data registers / address registers); and the task switcher including a memory management apparatus coupled to the input and the memory; and responsive to the instruction information indicating a return instruction for moving the return address from the storage location to the register (via performing a context switch thereby returning to a previous CSA) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28). Fleck also teaches the current context resides in the general purpose and program state registers of the processor and to save the current context and create a new one, the following steps can be performed by the processor hardware: a context save area is taken from the free list, the current context is stored into the save area, the save area is added to the head of the previous context list wherein the steps are performed in connection with a function call, or with taking a trap or interrupt thereby allowing the called function or interrupt or trap handler free to modify the general registers and other processor state without destroying the context of the calling function or interrupt task (col. 1, line 60 – col. 2, line 6). To switch back from a called function or trap or interrupt handler and switch back to the previous context, the following steps are performed by the processor handler: the save area at the head of the previous context list is removed from that list, the current context is loaded from the save area just removed from the previous context

list, and the save area is added to the free context save area list. These steps are performed as part of the function return instruction, or the instruction to return from an interrupt or trap handler. FLECK also teaches on figure 4 that the PCX has three contexts which would be associated with three previously executing tasks / interrupt handlers. Therefore, the teachings of FLECK teach an interrupt handler/task executing such that when it finishes it invokes a return operation and the context switch control unit switches back to the head of the PCX pointer, thereby switching context / loading the previously executing second task / interrupt handler, such that when it finishes it invokes a return operation and the context switch control unit switches back to the new head of the PCX pointer, thereby switching context / loading the now previously executing third task / interrupt handler wherein all of the tasks are different from one another (via one is a caller, one is the called, one is an interrupt handler and some of the tasks execute in different regions).

As to claim 2, FLECK teaches the selecting step includes the task switching function selecting a first pointer (address / pointer) that points to a first area of memory (CSA) where the return address is stored (col. 5, lines 49-67; col. 6, lines 9-12; col. 6, lines 47-54).

As to claim 3, FLECK teaches the pointer selecting step includes updating a second pointer to point to the first pointer (via restoring a CSA from the list / read-write-modify operation) (col. 6, lines 21-30).

As to claim 4, FLECK teaches the updating step includes updating the second pointer from a status wherein the second pointer points to a third pointer to a status wherein the second pointer points to the first pointer, and wherein the third pointer points to a second area of memory (via restoring a CSA from the list / read-write-modify operation) (col. 6, lines 21-30).

As to claim 5, FLECK teaches a return address corresponding to the first data processing task is stored in the second area of memory (via the second portion of the CSA which stores a pointer to another CSA / via following the list to the previous CSA) (col. 5, lines 49-61; col. 7, lines 10-28).

As to claim 6, FLECK teaches the task switching function storing the third pointer (via context switching functions thereby restoring CSAs from the list) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28).

As to claim 7, FLECK teaches the task switching function deselecting a return address (pointer / address) corresponding to the first data processing task (via restoring from the linked list) (col. 2, lines 7-21).

As to claim 8, FLECK teaches saving a return address (pointer / address) corresponding to the first data processing task (via switching context based on call) (col. 4, lines 10-30), and executing the saving step in parallel with the call executing step (col. 4, lines 49-55).

As to claim 12, FLECK teaches memory includes a second storage location for storing a first pointer (address / pointer) which points to a first area of the memory (CSA) that includes the first storage location, the memory management apparatus responsive to the instructions information for selecting the first pointer (via chaining addresses of CSAs to the list or using the list restore chained addressed CSAs on the list) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28).

As to claim 13, FLECK teaches the memory management apparatus includes a memory manager for maintaining a second pointer (address / pointer), the memory manager responsive to the instruction (context switch / restore) information for updating the second pointer to point to the first pointer in the memory (via updating the listed chained addressed CSAs after appending or removing CSA's to the list / read-write-modify operation) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-30; col. 6, line 47 – col. 7, line 28).

As to claim 14, FLECK teaches the memory manager is operable for updating the second pointer from a status wherein the second pointer points to a third pointer stored at a third location in the memory to a status wherein the second pointer points to the first pointer, and wherein the third pointer points to a second area of the memory (via context switching functions thereby storing CSAs to the list) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28).

As to claim 15, FLECK teaches the second area of the memory (part of CSA / part of list element) includes a fourth storage location which stores therein a return address (address / pointer) corresponding to the first data processing task (CSA for function / interrupt handler) (via switching context based on call) (col. 4, lines 10-30).

As to claim 16, FLECK teaches the memory manager is responsive to the instruction information for storing the third pointer (address / pointer of subsequent listed CSAs) in the third location of the memory (via context switching based on a call to store the address of CSA on list and link with the previous CSA) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28).

As to claim 19, FLECK teaches the register is a program counter register (register) (col. 3, lines 21-32).

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claim 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over FLECK (U.S. Patent 6,128,641).

As to claim 18, FLECK substantially discloses the invention above. However, FLECK does not teach the switcher includes TriCore data processor architecture. Official Notice is taken in that a Tri-Core data processor architecture is well known in the art and therefore would be obvious to one skilled in the art that the system of FLECK is implemented in a Tri-Core architecture in order to reduce execution overhead.

5. Claims 9 and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over FLECK (U.S. Patent 6,128,641) in view of Applicant's Admitted Prior Art (APA).

As to claim 9, FLECK substantially discloses the invention above. However, FLECK does not teach the data processing task being a host task, disk task, or servo task. APA teaches that a data processing task is one of a host task, a disk task and a servo task of an optical drive control system (pg. 1, line 21 – 2, line 1). It would be obvious to one skilled in the art at the time of the invention to combine the teachings of FLECK with the teachings of APA in order to switch firmware tasks more quickly than

conventional microprocessors and microcontrollers (col. 4, lines 38-41; col .1, lines 27-31).

As to claim 20, FLECK substantially discloses the invention above. However, FLECK does not teach the data processing task being a host task, disk task, or servo task. APA teaches that a data processing task is one of a host task, a disk task and a servo task of an optical drive control system (pg. 1, line 21 – 2, line 1). It would be obvious to one skilled in the art at the time of the invention to combine the teachings of FLECK with the teachings of APA in order to switch firmware tasks more quickly than conventional microprocessors and microcontrollers (col. 4, lines 38-41; col. 1, lines 27-31).

Response to Arguments

6. Applicant's arguments filed July 9, 2007 have been fully considered but they are not persuasive. Applicant argues that the language of the first, second, and third illustrates different task and that there is no teaching in Fleck that teaches the calling task, the called task, and the interrupt handler being different task wherein one of the calling task or the called task is the interrupt handler or that the tasks are subtasks of an overall task as being submitted without basis. The examiner disagrees. The term used in the claims is a first task, a second task, and a third task. There is no clear understanding of how first, second, and third regarding a task illustrates the difference in the tasks. The interpretation of first task, second task, third task is unlimiting, some of

which are provided below. One interpretation is that the first, second, and third task are subtask of an overall task, such that the switching performs switching between the different subtask. Another interpretation is a combination of the tasks are on part of the same task. Another is that all are separate executable operations from one another and are not directly related, i.e. task A is a speadsheet task, task B is a word processing task, task C is a mail task. M.P.E.P. 2111, allows the examiner to interpret the claims in the broadest sense consistent with the specification and without reading in any limitations of the specification. There is no language in the specification or the claims that would not allow for any of the cited interpretations, including the task being subtasks of an overall task. Fleck teaches two types of task, software managed task and interrupt service routines (col. 3, lines 5-7). Software management task are user task and execute in user mode whereas interrupt service task are expected to execute in supervisor mode (col. 3, lines (15-20). Fleck states that when a function call is made the calling routine has a context that must be saved and then restored in order to resume the caller's execution after return from the function (col. 4, lines 11-13). Similarly, when an interrupt occurs asynchronously, the context saved must include all registers that the interupted task might be using (col. 4, lines 10-17). Therefore, when the called function experiences an interrupt, its registers must be saved such that the interrupt service routine is switched in and executed. Applicant's argument that one of the calling or called routines is the ISR is improper, because the ISR is in the supervisor mode, e.g. lower protection domain, from the user tasks. Therefore, the reference teaches three task in the broadest sense.

Applicant argues that there is no requirement that the ISR have to execute in supervisor mode whereas the reference details that the software managed tasks are assumed o execute in user mode and ISR's are expected to execute in supervisor mode. The examiner disagrees. The claim language used is different tasks. As stated above, different has a wide range of meanings. Its not just limited to the place of execution but how it executes also. The differences in the task can be that one is the caller, one is the called, and one handles interrupts. The difference in the task can also be as used before, where they execute, e.g. in supervisor mode and in user mode. By expecting the ISR to execute in supervisor mode, the reference is inherently teaching that the reference requires the ISR to execute in supervisor mode. There is no language in the claims that detail any other area where the ISR execute. The remaining arguments in this section deal with where the other tasks may be executing. The examiner applies the same rationale as detailed above, regarding the meaning of different in maintaining the rejection.

Applicant remaining arguments apply that Fleck either singly or in combination does not teach returning execution to a previous task, i.e. second, third, etc., and at best teaches a context save area in front of the previous context list getting written to. The examiner disagrees. Fleck is concerned with the saving of task information when switching between tasks, i.e. when calling a software routine or returning from a function to its caller (col. 1, lines 28-44). Fleck teaches to save a current context, a context save area (CSA) is taken from the free list, the current context is stored into the save area, and the save area is added to the head of the previous context list (col. 1, lines 60-66).

The saving of a context is performed in connection with a function call, or with taking a trap or interrupt, thereby leaving the called function or interrupt or trap handler free to modify the general registers and other processor state without destroying the context of the calling function or interrupt task (col. 2, lines 1-6). To exit a called function or trap or interrupt handler and switch back to the previous context, the processor performs the steps of the save area at the head of the previous context list is removed from that list, the current context is loaded from the save area just removed from the previous context list, the save area is added to the free context save area list (col. 2, lines 7-15). These steps are performed as part of the function return instruction, or the instruction to return from an interrupt or trap handler (col. 2, lines 16-18). From at least the above recitations, Fleck teaches returning execution to a previous task whose context was saved, by executing a return operation that takes the head of the previous context list and loads its values. By the amount of times this is executed, there are different values. Therefore, Fleck adequately teaches the limitations of the claims as written and therefore the rejection is maintained.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (571) 272-3759. The examiner can normally be reached on Monday-Friday, 8:30 a.m. - 5:00 p.m..

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

September 17, 2007

LEWIS A. BULLOCK, JR.
PRIMARY EXAMINER